Внедряем скоростное мутационное тестирование



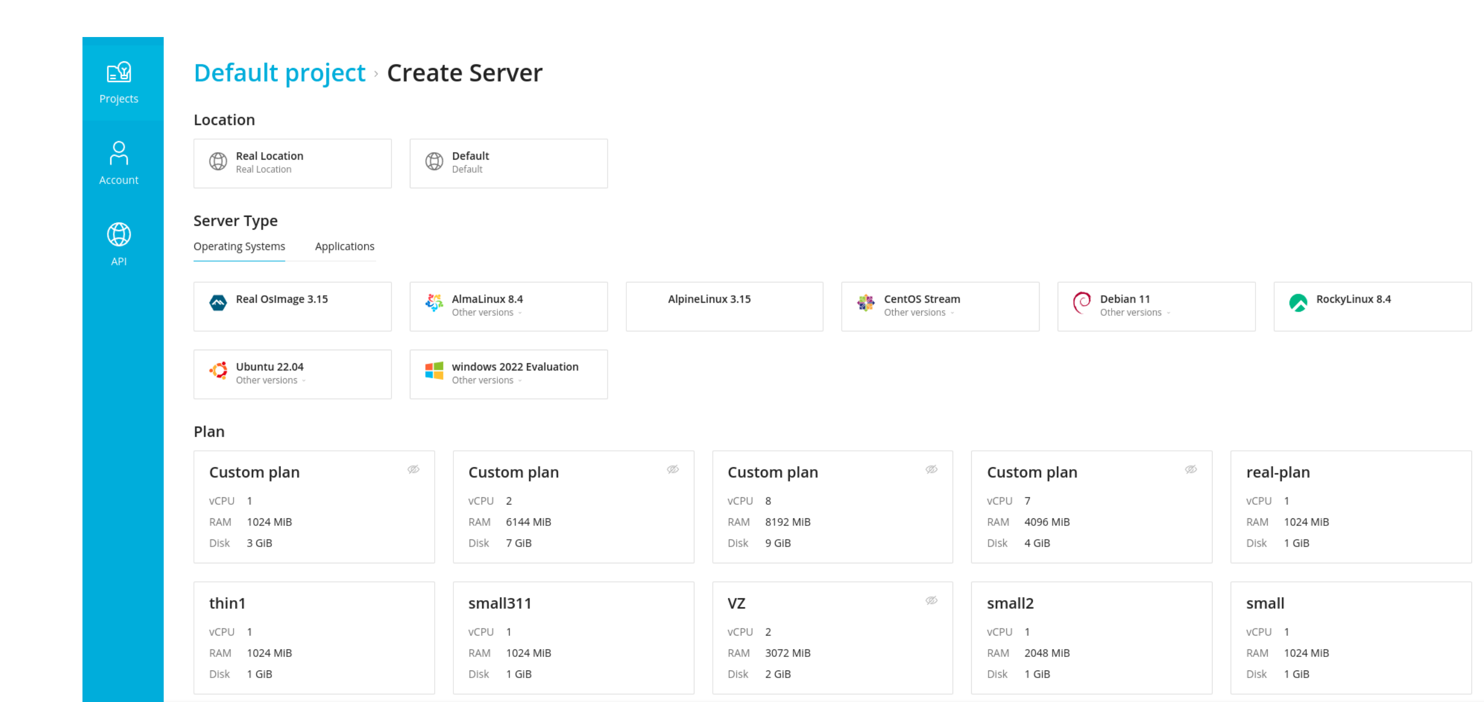


Станислав Вожов
Software Developer
WebPros, Solus Team



SolusVM 2.0

- Коробочное web-приложение
- Laravel, Go, React, PostgreSQL
- PHPUnit, Jest, React testing library, Cypress
- 4 fullstack-разработчика
- 4 года



Количество строк кода: 280000+
Количество тестов: 6000+
Code Coverage: 90%



Как проверить эффективность наших тестов?



Path и branch coverage

Yes
Action 1
Action 2
Start
Decision
Finish
No
Action 3



Path и branch coverage

vendor/bin/phpunit --path-coverage

+ позволяет оценить все пути выполнения функции
- время выполнения (один прогон занимал бы больше 24 часов)



Path и branch coverage. Почему нет?

- Выполняется ОЧЕНЬ долго (подтверждено авторами), даже только на изменённых файлах в PR.
- Хотим автоматически запускать тесты до закрытия PR. Запуск по команде в PR или отдельно от PR не рассматриваем.

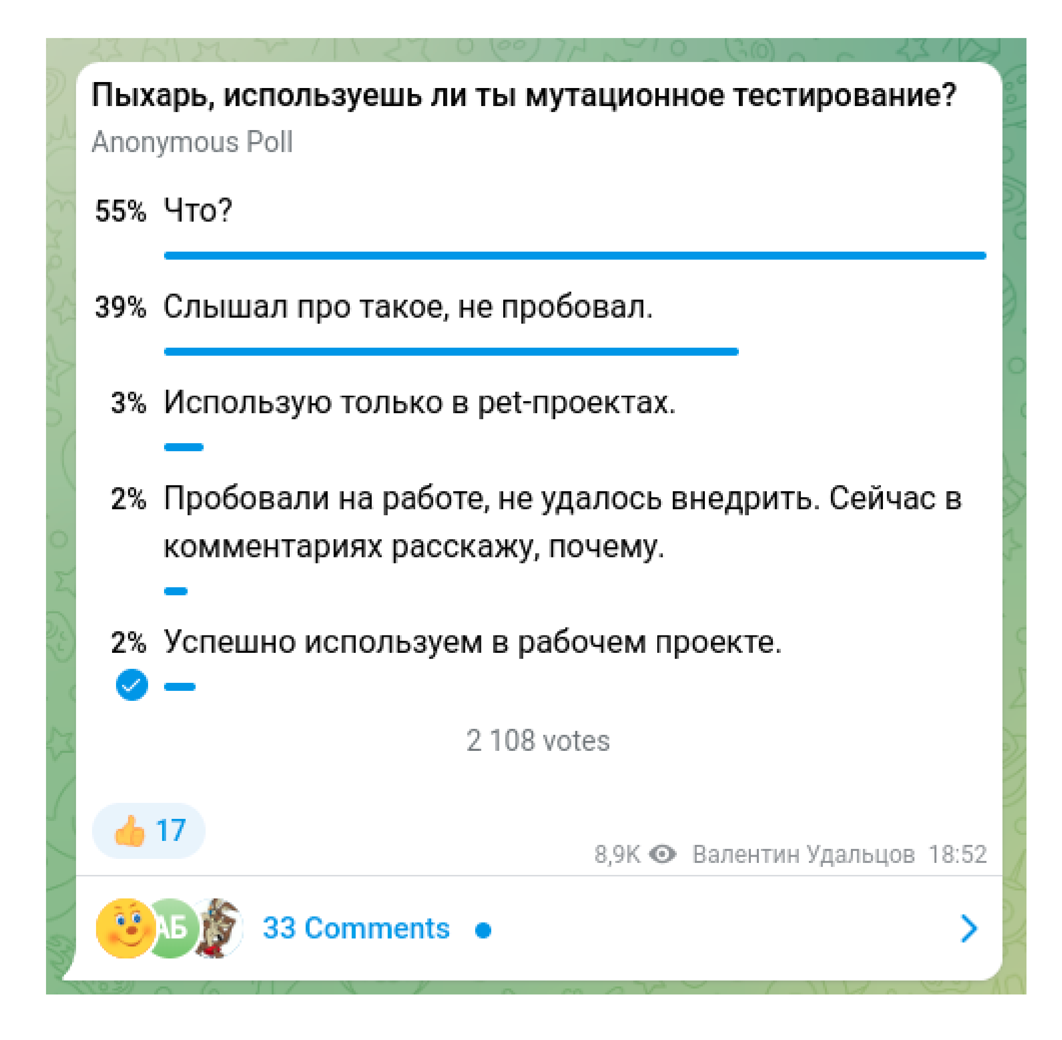


Мутационное тестирование

```
public function hasErrors(): bool
{
-    return count($this->errors) > 0;
+    return count($this->errors) >= 0;
}
```




Мутационное тестирование





Infection — Mutation Testing framework

- https://github.com/infection/infection
- Maks Rafalko — Mutation Testing in PHP
- Владимир Янц (Badoo) — Мутационное тестирование в PHP



Кому может быть полезно мутационное тестирование?

- Нет тестов, но хочется начать их писать
- Много сложной логики или расчётов
- Часть приложения связана с деньгами
- Много тестов и высокий code coverage



Примеры внедрения Infection в компании

Успешно удалось внедрить в:

- небольшой сервис с Code Coverage 85% и 800+ тестов (Infection был добавлен за 1.5 дня)

Не удалось внедрить в:

- проект с 1M+ строк кода
- проект средних размеров (250K строк кода)



Внедрение Infection



Запуск мутационных тестов на весь код



Запуск мутационных тестов только на изменённых файлах

- infection --git-diff-filter=AM
- infection --git-diff-lines
- infection --filter=Mailer.php,Foobar.php



Выбор характеристики в качестве барьера



Mutation Score Indicator (MSI)

TotalDefeatedMutants = KilledCount + TimedOutCount + ErrorCount

MSI = (TotalDefeatedMutants / TotalMutantsCount) * 100



Covered Code MSI

TotalCoveredByTestsMutants = TotalMutantsCount - NotCoveredByTestsCount

CoveredCodeMSI = (TotalDefeatedMutants / TotalCoveredByTestsMutants) * 100



Внедрение Infection. Сообщение об успешной сборке

Jenkins APP 3 days ago

Solus: **build 1.3.0-24157** tested successfully!
Latest commit: 1db773
Job: solusvm/ci-solusnxt-build
Jenkins node: d63.qa.plesk.ru running on 10.54.0.15
PHP
Cover:
 Classes: 67.76% (643/949)
 Methods: 83.35% (2918/3501)
 Lines: 89.44% (14547/16265)

Mutation testing (Log):
 MSI: 72.68%
 Mutation Code Coverage: 90.55%
 Covered Code MSI: 80.26%

JS cover:
 Statements: 82.98% (9907/11939)
 Branches: 72.31% (3126/4323)
 Functions: 78.35% (3178/4056)
 Lines: 84.76% (8916/10519)



Внедрение Infection. Сообщение в PR

```
56 57     public function deleting(ComputeResourceVm $vm): void
57 58     {
58    -       Usage::collect(UsageResourceDTO::forUptime($vm));
59    -       Usage::collect(UsageResourceDTO::forBackup($vm));
   59 +       UsageCollectorService::collect(UsageResourceDTO::forUptime($vm));
   60 +       UsageCollectorService::collect(UsageResourceDTO::forBackup($vm));
```

Git Bot 1 week ago

☑ most interesting escaped `MethodCallRemoval` mutant:

```
+++ New
@@ @@
     public function deleting(ComputeResourceVm $vm) : void
     {
         UsageCollectorService::collect(UsageResourceDTO::forUptime($vm));
-        UsageCollectorService::collect(UsageResourceDTO::forBackup($vm));
+
         $vm->additionalIps()->each(static function (Ip $ip) : void {
             UsageCollectorService::collect(UsageResourceDTO::forAdditionalIp($ip));
         });
```

Reply



Внедрение Infection. Сообщение в PR

Git Bot 3 days ago NEW

☑ Infection stats:

```
Total mutants: 702
Killed: 683
Not Covered: 5
Escaped: 14
Time outed: 0
Errored: 0
MSI: 97.29%
Mutation Code Coverage: 99.29%
Covered Code MSI: 97.99%
```




Минусы Infection



Невозможность запускать на весь проект



Минусы Infection. Бесполезные мутации

```
public static function incrementHours(UsageResourceDTO $usageResourceDTO, int $hours): void
public static function recollect(UsageResourceDTO $usageResourceDTO): void
```

Git Bot 15 Jun 2022

most interesting escaped `PublicVisibility` mutant:

```
+++ New
@@ @@
         $usedHours = self::getDiffInHours($now, $usage->started_at);
         $usage->update(['tokens' => $usageResourceDTO->tokensPerHour * $usedHours, 'quantity' =>
$usedHours]);
     }
-    public static function recollect(UsageResourceDTO $usageResourceDTO) : void
+    protected static function recollect(UsageResourceDTO $usageResourceDTO) : void
     {
```




Минусы Infection. Эквивалентные мутаторы

```
if (!$usageResourceDTO->shouldCollect && $usage->started_at && !$usage->ended_at) {
```

Git Bot 15 Jun 2022

- [x] most interesting escaped `LogicalAnd` mutant:

```
+++ New
@@ @@
                return;
            }
            // Stop collecting usage
-           if (!$usageResourceDTO->shouldCollect && $usage->started_at && !$usage->ended_at) {
+           if (!$usageResourceDTO->shouldCollect && $usage->started_at || !$usage->ended_at) {
                self::finishUsageCollecting($usage);
                return;
            }
```




Результаты внедрения

- Мутационные тесты, выполняются на каждое изменение PHP-кода
- На +3 минуты в среднем увеличилось время прохождения сборок, что для нас не критично
- Добавлен подсчёт MSI на каждом PR: барьер 85%
- Отправляются сообщения в PR о выживших мутантах



Результаты использования

- Не нужно много внимания уделять тестам во время code review
- Весь новый код покрывается тестами
- Места со сложной логикой или расчётами протестированы лучше



Результаты использования

- Во время рефакторинга легко найти плохо протестированные места
- Некоторые участки кода написаны лучше благодаря Infection



Сколько дал нам Infection в деньгах?



Планы на будущее



Использовать свои кастомные мутаторы



PR в Infection

Support custom mutators #1686

Open **vss414** wants to merge 1 commit into `infection:master` from `vss414:feature/support-custom-mutators`

Conversation 3 | Commits 1 | Checks 10 | Files changed 23

vss414 commented on Apr 29

Infection supports only a set of Mutators which are based on AST and PHP-Parser project. In such situation library users can't use their own mutators (which may be useful in their projects but doesn't fit to be a part of the library).
This PR adds the ability to use custom mutators in projects that use the infection library.
Custom mutators should implement `Infection\Mutator\Mutator` interface.

To use custom mutators, user must add them to the mutator list in the infection config. User has to use the class name with the namespace of the custom mutator as a name of mutator in the config. An example of using a custom mutator in the infection config:



Добавить мутационное тестирование для JavaScript (Stryker)





Итого



Внедрить несложно, если уже есть тесты



Может выполняться быстро

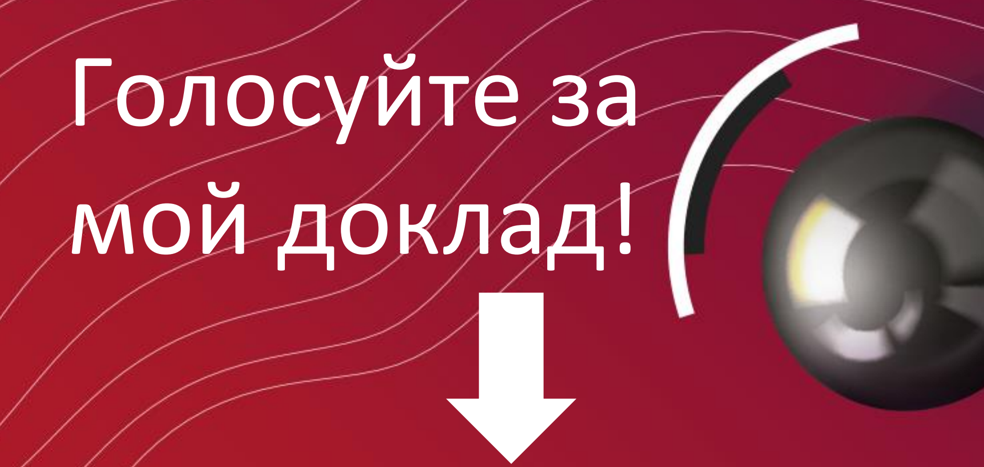


Показывает эффективность инвестиций в тестирование и качество самого кода



Итого:

- Внедрить несложно, если уже есть тесты
- Может выполняться быстро
- Показывает эффективность инвестиций в тестирование и качество самого кода





Станислав Вожов
@svozhov

